HOST-INITIATED DATA RECONSTRUCTION FOR IMPROVED RAID READ OPERATIONS

RELATED APPLICATIONS

[1] This application incorporates by reference commonly assigned and co-pending U.S. Patent Application Number _________ , IBM Docket Number TUC9-2003-0163US1, entitled RECOVERING TRACK FORMAT INFORMATION MISMATCH ERRORS USING DATA RECONSTRUCTION, filed on even date herewith.

TECHNICAL FIELD

[2] The present invention relates generally to RAID storage systems and, in particular, to improving the integrity of read operations.

BACKGROUND ART

[3] In a non-RAID computer system, if a disk drive fails, all or part of the stored customer data may be permanently lost (or possibly partially or fully recoverable but at some expense and effort). Although backup and archiving devices and procedures may preserve all but the most recently saved data, there are certain applications in which the risk of any data loss and the time required to restore data from a backup copy is unacceptable. Therefore, RAID ("redundant array of inexpensive disks") storage subsystems are frequently used to provide improved data integrity and device fault tolerance. If a drive in a RAID system fails, the entire data may be quickly and inexpensively recovered.

[4] There are numerous methods of implementing RAID systems. Such methods are commonly known in the industry and only a few will be described, and only generally, herein. A very basic RAID system, RAID level 1, employs simple mirroring of data on two parallel drives. If one drive fails, customer data may be read from the other. In RAID level 2, bits of a data word are written to separate drives, with ECC (error correction code) being written to additional drives. When data is read, the ECC verifies that the data is correct and may correct incorrect data caused by the failure of a single drive. In RAID 3, data blocks are divided and

written across two or more drives. Parity information is written to another, dedicated drive. Similar to RAID 2, data is parity checked when read and may be corrected if one drive fails.

[5] In RAID level 5, data blocks are not split but are written block by block across two or more disks. Parity information is distributed across the same drives. Thus, again, customer data may be recovered in the event of the failure of a single drive. RAID 6 is an extension of RAID 5 and allows recovery from the simultaneous failure of multiple drives through the use of a second, independent, distributed parity scheme. Finally, RAID 10 (or 1-0) combines the mirroring of RAID 1 with data striping. Recovery from multiple simultaneous drive errors may be possible.

[6] The types of errors from which traditionally implemented RAID systems may recover only include those which the RAID controller detects. One common error detectable by the controller is a media error. In certain systems developed and sold by International Business Machines (IBM®), another controller-detectable error is one which is detectable through the use of block LRCs appended to each sector. ("LRC" refers to a longitudinal redundancy check word attached to a block of data and used to ensure that the block is delivered error-free.)

[7] However, other errors may not be detectable by a RAID controller. For example, when the LRCs are generated across multiple sectors, the RAID controller may not able to detect certain errors. The controller may also not be able detect errors in sequence numbers embedded in the data. Another example of an error which may not be detectable by the RAID controller can occur when data is not actually written to one of the drives but the RAID controller, not detecting the failure, directs that the correct parity be written.

[8] While the host or client may be able to detect some errors which the RAID controller does not, there is currently no recovery procedure available. Thus, a need exists to permit recovery of data errors which are not detectable by the RAID controller.

SUMMARY OF THE INVENTION

[9] The present invention provides method, system and computer program product to improve the reliability of data transfers from RAID systems. In one embodiment, a command is transmitted from a host device to a RAID controller to read a block of data from an array of storage drives. The block of data is obtained by the RAID controller from the drives and transmitted to the host. The host determines whether an error is present in the data. If so, the host transmits a second command to the RAID controller to re-read the data in a reconstruct mode. The RAID controller reconstructs the block of data and transmits it to the host.

[10] The first command may include an instruction directing the RAID controller to use a first of a plurality of reconstruct read algorithms and the second command may include an instruction directing the RAID controller to use a second of the plurality of reconstruct read algorithms.

[11] In a further embodiment, the host detects errors in the received reconstructed data. If an error is detected, the host transmits a third command to the RAID controller to re-read the data in a second reconstruct mode.

[12] In still a further embodiment, the first command may include an instruction directing the RAID controller to read a first of two copies of the data and the second command may include an instruction directing the RAID controller to read a second of the two copies.

[13] Additionally, an indication of an error may be provided whereby a faulty drive may be replaced.

BRIEF DESCRIPTION OF THE DRAWINGS

[14] Fig. 1 is a block diagram of a generic RAID system in which the present invention may be implemented; and

[15] Fig. 2 is a flow chart of an implementation of the present invention.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

[16] Fig. 1 is a block diagram of a generic RAID system 100 in which the present invention may be implemented. The system 100 includes a RAID controller 110 coupled to a host or client 120. The controller 110 is also coupled to the RAID disk

array 130 itself which includes two or more disk drives. The host 120 includes a processor 124 which executes routines and issues read and write commands to the RAID controller 110. The RAID controller 110 also includes a processor 114 which processes commands received from the host 120 and executes RAID drive management routines. The controller 110 may also include a cache 116 for temporary storage of recently or often accessed customer data.

[17] Referring also to the flow chart of Fig. 2, the host 120 issues a read command to the RAID controller 110 to retrieve specified customer data (step 200). The controller 110 determines the physical location of the data on the drives 130 or in the cache 116 (step 202). After the data is located, it is retrieved (step 204) and verified by the controller 110 (step 206). If an error is detected (such as might be caused by a medium error), the controller 110 "reconstructs" the data using the appropriate RAID algorithm (step 208). In the case of RAID level 1 or 10, the algorithm includes reading the data from another drive. In the case of other RAID levels, the algorithm includes using ECC, parity or another scheme to actually reconstruct the desired data. As used herein, the term "algorithm" will refer to any of these methods and the term "reconstruct" will refer to the process of applying of any of these methods. In the event the reconstruction fails (not shown), the process ends. When the data has been verified or reconstructed, it is transmitted to the host 120 (step 210).

[18] As noted above, however, there are certain type of errors which might escape detection by the RAID controller 110. Consequently, the host 120 also attempts to detect errors in the data received from the controller 110 (step 212). If no errors are detected, the process of the present invention ends (step 214). However, if an error is detected, the host 120 transmits another command to the controller 110 (step 216) to reconstruct the desired data. When this second command is executed, the RAID controller 110 applies the appropriate algorithm (step 218) to reconstruct the data. The controller 110 is not permitted to resend the same, faulty, data, whether from the drives 130 or from the cache 116. The controller 110 then sends the reconstructed data back to the host 120 (step 220) where it is again verified (step 222).

[19] For example, when a RAID 1 or 10 system is used, one of the drives is considered to be the primary drive and the other is considered to be the secondary. However, the designations are typically arbitrary and a conventional read command may return data from either drive or from a combination of the two. In one embodiment of the present invention, the first read command transmitted by the host 120 to the controller 110, may include an instruction to read a specified drive (for example, FF_ReadPrimary). The second command, if required, may then include an instruction to read another specified drive (for example, FF_ReadSecondary). Thus, assuming that the second drive contains different and correct data, it can be assured that incorrect data will not be re-transmitted from the controller 110 to the host 120. It will be appreciated that more advanced RAID systems may be accommodated by the present invention by employing corresponding additional commands.

[20] In a variation of the foregoing procedure, the host 110 may also compare the two sets of read data and determine if either is correct.

[21] As noted above, a RAID 6 system achieves a high degree of fault tolerance through the use of two (or more) RAID algorithms. In another embodiment of the present invention, the first read command transmitted by the host 120 to the controller 110, may include an instruction to apply a specified one of the algorithms. The second command, if required, may then include an instruction to read the data using another algorithm. Thus, if the second algorithm results in correct data, it can be assured that incorrect data will not be re-transmitted from the controller 110 to the host 120.

[22] When a RAID 5 system is used, another embodiment of the present invention may be implemented. In a 3+P RAID 5 system, data blocks are written to three drives and parity for the three blocks is written to a fourth drive. For purposes of this example, the blocks of data may be labeled A, B and C; the parity drive normally will be generated from A xor B xor C. If a data block D is intended to be written to the second drive, it should replace block B. However, occasionally the write operation may fail without the RAID controller 110 detecting the failure. Consequently, the parity will be updated as A xor D xor C while block B remains intact on the second

drive. Thus, a read command will return blocks A, B and C, not A, D and C. Such an error may not be detectable by the controller 110. The host 120, however, may detect the error and command the controller 110 to reconstruct the data on the second drive using the parity. The host 120 would then correctly receive blocks A, D and C.

[23] Because it is important to prevent future errors as well as correct for existing drive failures, an error log may be recorded and analyzed to determine which methods of reading the customer data result in obtaining the correct data. The host 120 may also use the error log to isolate a failure in a drive 130. A faulty drive 130 may be replaced after a predetermined number of failures. More likely, it will be desired to replace a drive after the first failure to reduce the probability of a future failure of the same drive and the attendant risk of having two drives fail simultaneously. While some RAID levels are designed to allow recovery from a multi-drive failure, others levels are not and a multi-drive failure could result in the loss of data.

[24] The objects of the invention have been fully realized through the embodiments disclosed herein. Those skilled in the art will appreciate that the various aspects of the invention may be achieved through different embodiments without departing from the essential function of the invention. The particular embodiments are illustrative and not meant to limit the scope of the invention as set forth in the following claims.